Research Article

A Study on Distributed Optimization over Large-Scale Networked Systems

Hansi K. Abeynanda and G. H. J. Lanel

1Mathematics Unit, Faculty of Humanities and Sciences, Sri Lanka Institute of Information Technology, Malabe, Sri Lanka
2Department of Mathematics, Faculty of Applied Sciences, University of Sri Jayewardenepura, Nugegoda, Sri Lanka

Correspondence should be addressed to Hansi K. Abeynanda; kavindika.a@sliit.lk

Received 8 February 2021; Revised 2 April 2021; Accepted 7 April 2021; Published 29 April 2021

Distributed optimization is a very important concept with applications in control theory and many related fields, as it is high fault-tolerant and extremely scalable compared with centralized optimization. Centralized solution methods are not suitable for many application domains that consist of large number of networked systems. In general, these large-scale networked systems cooperatively find an optimal solution to a common global objective during the optimization process. Thus, it gives us an opportunity to analyze distributed optimization techniques that is demanded in most distributed optimization settings. This paper presents an analysis that provides an overview of decomposition methods as well as currently existing distributed methods and techniques that are employed in large-scale networked systems. A detailed analysis on gradient like methods, subgradient methods, and methods of multipliers including the alternating direction method of multipliers is presented. These methods are analyzed empirically by using numerical examples. Moreover, an example highlighting the fact that the gradient method fails to solve distributed problems in some circumstances is discussed under numerical results. A numerical implementation is used to demonstrate that the alternating direction method of multipliers can solve this particular problem, by revealing its robustness compared with the gradient method. Finally, we conclude the paper with possible future research directions.

1. Introduction

Optimization is a mathematical discipline which determines the best possible solution corresponding to the optimum performance of a quantitatively well-defined system. The theory of optimization has been established as a desirable tool that is used in a wide range of disciplines, such as automatic control systems, estimation and signal processing, communications and networks, electronic circuit design, data analysis and modeling, statistics, and finance [1–3]. In the recent study [4], the novelty search, a tool that is used in evolutionary and swarm robotics was developed for the use of global optimization. Formally, a mathematical optimization problem can be posed as follows:

\[
\begin{align*}
\text{minimize} & \quad f_0(x) \\
\text{subject to} & \quad x \in C,
\end{align*}
\]

where \( f_0 \) is a real-valued objective function of the decision variables \( x \in \mathbb{R}^n \) and \( C \subseteq \mathbb{R}^n \).

However, in reality, it may be difficult or not possible to find analytic solutions to certain optimization problems. As a result, iterative methods that provide approximate solutions have been introduced by researchers. Algorithms that are used to solve optimization problems have been extensively analyzed mainly under centralized and decentralized architectures [5, 6]. Centralized solution methods are not suitable for many communication networking problems such as large-scale and data-intensive problems that demand distributed solutions. Consequently, the application of distributed optimization techniques where subsystems coordinate to find a solution to the original problem is of utmost importance. Intranets, the Internet, telecommunication networks, aircraft control systems, sensor networks, and electronic banking are some important examples for
distributed systems. These systems consist of a large number of smaller subsystems, and they integrate together to reach an optimal status of the process. This optimal status of process in large-scale networked systems needs to be achieved without incurring errors and exceeding already set time limits for expected outcomes. Therefore, the study of well-established theoretical concepts together with empirical implementations on distributed optimization is critical. This gives us an opportunity to analyze currently existing distributed techniques and methods. In general, we may have many subsystems in a distributed optimization setting. We consider the following optimization problem with five subsystems as an example to provide a deeper explanation of distributed optimization:

\[
\begin{align*}
\text{minimize} & \quad f_1(x_1, y) + f_2(x_2, y, r) \\
& \quad + f_3(x_3, y) + f_4(x_4, y, z) + f_5(x_5, r, z) \\
\text{subject to} & \quad x_1 \in \mathcal{X}_1, x_2 \in \mathcal{X}_2, x_3 \in \mathcal{X}_3, x_4 \in \mathcal{X}_4, \\
& \quad x_5 \in \mathcal{X}_5, y \in \mathcal{Y}, r \in \mathcal{R}, z \in \mathcal{Z},
\end{align*}
\]

where \(\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3, \mathcal{X}_4, \mathcal{X}_5, \mathcal{R},\) and \(\mathcal{Z}\) are subsets of \(\mathbb{R}^n\). In this problem, we can observe that there are three complicating variables \(y, r,\) and \(z\). The variable \(y\) is shared by subsystems 1, 2, 3, and 4, the variable \(r\) is shared by subsystems 2 and 5, while the variable \(z\) is shared among subsystems 4 and 5. Figure 1 shows the associated decomposition structure of (2), and the related distributed problem can be stated as follows:

\[
\begin{align*}
\text{minimize} & \quad f_1(x_1, y_1) + f_2(x_2, y_2, r_1) + f_3(x_3, y_3) \\
& \quad + f_4(x_4, y_4, z_1) + f_5(x_5, r_2, z_2) \\
\text{subject to} & \quad y_1 = y_2 = y_3 = y_4, r_1 = r_2, z_1 = z_2.
\end{align*}
\]

Here, we can observe that problem (3) is minimized by multiple users cooperatively. Hence, a distributed method is required to find a solution.

Many networked systems cannot communicate exact information between subsystems due to unavoidable errors that may occur as a result of limited communication bandwidths and sometimes due to measurement errors [7, 8]. Therein lies the importance of analysing quantized distributed methods in real life situations [9–13]. Although many quantized distributed methods have been analyzed, deeper investigation of quantization methods is still required.

We present the outline of our paper as follows. In Section 2, we discuss the preliminaries related to distributed optimization and primal and dual decomposition. Section 3 provides a general literature review on currently existing well-known distributed optimization methods. Next, in Sections 4, 5, and 6, we discuss the gradient method, the subgradient method, and the alternating direction method of multipliers (ADMM), respectively. In those sections, we discuss the theoretical concepts of the relevant methods as well as previous studies performed on them. In Section 7, we continue our discussion on distributed optimization with noise to emphasize the importance of involvement of error in distributed optimization methods. In Section 8, we provide our numerical results to discuss the convergence of aforementioned distributed methods. Finally, in Section 9, we conclude our paper with possible future research directions.

2. Preliminaries

In this section, we discuss the concept of distributed optimization and we introduce primal decomposition and dual decomposition, which play an important role in distributed optimization. Our introduction on primal decomposition and dual decomposition is mainly inspired by the lecture notes on decomposition methods by boyd et al. [14]. Throughout the paper, we will use following notations.

**Notation.** We let \(\mathbb{R}, \mathbb{R}^n,\) and \(\mathbb{R}^n_+\) represent the set of real numbers, \(n\)-dimensional Euclidean space, and positive orthant in \(n\)-dimensional Euclidean space, respectively. For \(x \in \mathbb{R}^n, \|x\|\) denotes the Euclidean norm and \([x]_n\) denotes the projection of \(x\) on to the set \(\mathcal{X} \subseteq \mathbb{R}^n\). The set of \(n \times m\) matrices is denoted by \(\mathbb{R}^{n \times m}\). The transpose of a matrix \(A\) is given by \(A^T\). \(\nabla f\) represents the gradient of a scalar valued function \(f\).

2.1. Distributed Optimization. Distributed optimization is an optimization process that is used in networked systems with a large number of users. This process enables the system to solve a global problem cooperatively even if there is no central controller available in the system. When compared with centralized techniques, distributed optimization has many considerable advantages. In distributed algorithms, nodes or users in the network share information only with necessary parties. This fact improves cyber security and reduces communication cost. Furthermore, distributed techniques have the ability to handle problems even if the problem size is very large. These techniques also have the potential to increase the solution speed [15].

Distributed optimization algorithms solve large-scale and data-intensive problems in a wide range of application areas such as communications [16–19], electricity grid [20, 21], large-scale multiagent systems [22, 23], smart grids, wireless sensor networks [24], and statistical learning. Zhang and Sahraei-Ardakani have developed a fully distributed DC optimal power flow method that incorporates flexible transmission and discussed the effect of communication limitations on the convergence properties [25, 26]. In [27],
authors have presented a study on finite-time consensus
opinion dynamics and studied an application to distributed
optimization over digraph.

Many distributed optimization algorithms are built on
decomposition methods. Decomposition is an interesting
approach to solving a global problem by breaking it up into
smaller subproblems and solving each of them separately.
These subproblems get solved either in parallel or sequen-
tially [6, 14, 28–30]. Decomposition in optimization appears
in early work on large-scale linear programs from 1960s [31].
The simplest decomposition structure is available in block
separable problems. For an example, a block separable
problem can be given as follows:

\[
\begin{align*}
\text{minimize} & \quad f_1(x_1) + f_2(x_2) \\
\text{subject to} & \quad x_1 \in \mathcal{X}_1, x_2 \in \mathcal{X}_2.
\end{align*}
\]

(4)

In this form, we can minimize \( f_1(x) \) and \( f_2(x) \) sepa-
rately in parallel and obtain the optimal value and optimal
solution. However, this method seems to be trivial and does
not seem to be an interesting task as many real life prob-
lems appear in a more complex form than this [14]. This problem
becomes more complicated and creates more interest when
the subvectors \( x_1 \) and \( x_2 \) are coupled. This situation can be
handled by primal decomposition and dual decomposition,
which are the most well-known decomposition methods
currently available.

2.2. Primal Decomposition. Primal decomposition deals with
complicating variables. Here, we consider a constrained
minimization problem that consists of \( m \) number of users as
follows:

\[
\begin{align*}
\text{minimize} & \quad f(x) = \sum_{i=1}^{m} f_i(x_i, y) \\
\text{subject to} & \quad x_i \in \mathcal{E}_i, i = 1, 2, \ldots, m, y \in \mathcal{Y},
\end{align*}
\]

(5)

where \( x = (x_1, x_2, \ldots, x_m, y), \mathcal{E}_i \subseteq \mathbb{R}^n, \mathcal{Y} \subseteq \mathbb{R}^n \), and \( f_i \)s rep-
resent real-valued objective functions of individual users.
Here, the variable \( y \) is called the complicating variable,
which complicates the system. When \( y \) is fixed, problem (5)
decomposes in to \( m \) smaller subproblems.

Subproblems are as follows:

\[
S_i(y) = \text{minimize } f_i(x_i, y).
\]

(6)

Then, the original problem (5) is equivalent to the problem

\[
\text{minimize } S(y) = \sum_{i=1}^{m} S_i(y),
\]

(7)

and this is called the master problem in primal decom-
position [14]. Next, the original problem (5) can be solved by
solving the master problem (7), using a distributed algo-
rithm under some well-defined assumptions on individual
primal objective functions \( f_i \)s.

2.3. Dual Decomposition. Here, we consider the same
problem (5) discussed under primal decomposition only with
two users. Then, we have the objective function as
\( f(x) = f_1(x_1, y) + f_2(x_2, y) \). Next, the problem can be
rearranged by introducing new variables \( y_1 \) and \( y_2 \) as follows
[14]:

\[
\begin{align*}
\text{minimize} & \quad f(x_1, x_2, y_1, y_2) = f_1(x_1, y_1) + f_2(x_2, y_2) \\
\text{subject to} & \quad y_1 = y_2, x_1 \in \mathcal{E}_1, x_2 \in \mathcal{E}_2, y_1, y_2 \in \mathcal{Y}.
\end{align*}
\]

(8)

According to this new arrangement, the objective function \( f \) is separable. Next, we can apply the decom-
position with its dual problem. The Lagrangian of (8) is given
by

\[
L(x_1, x_2, y_1, y_2, \lambda) = f_1(x_1, y_1) + f_2(x_2, y_2) + \lambda^T (y_1 - y_2).
\]

(9)

Next, the related dual function is given by

\[
g(\lambda) = \inf_{x_1 \in \mathcal{E}_1, x_2 \in \mathcal{E}_2, y_1, y_2 \in \mathcal{Y}} L(x_1, x_2, y_1, y_2, \lambda),
\]

(10)

which is accompanied with subproblems

\[
\begin{align*}
g_1(\lambda) & = \inf_{x_1 \in \mathcal{E}_1, y \in \mathcal{Y}} f_1(x_1, y_1) + \lambda^T y_1, \\
g_2(\lambda) & = \inf_{x_2 \in \mathcal{E}_2, y \in \mathcal{Y}} f_2(x_2, y_2) - \lambda^T y_2.
\end{align*}
\]

(11)

Then, the dual problem of (8) is given by

\[
\text{maximize } g(\lambda) = g_1(\lambda) + g_2(\lambda).
\]

(12)

This is called the master problem in dual decomposition.
This problem can be solved by using an iterative method
such as subgradient method, which will be discussed under
Section 5. Although we are able to solve the dual problem
and find dual optimal measures, we still cannot guarantee
that we can find primal optimal measures without intro-
ducing some acceptable conditions on the primal objective
function. For an example, if \( f_1 \) and \( f_2 \) are strictly convex,
then the primal variables \( x_1, x_2, y_1 \), and \( y_2 \) found by solving
two subproblems \( g_1 \) and \( g_2 \) are guaranteed to converge to
the optimal solution of the primal problem (8) [14].

3. A General Literature Review on Distributed
Methods for Solving Optimization Problems

In this section, we provide a general overview of currently
existing distributed optimization methods. A detailed
analysis will be given in later sections with more technical
details. Most of the existing studies done on distributed
optimization problems have been analyzed and related so-
lution methods have been discussed when the optimization
problem is convex. Convex optimization problems can be
solved very reliably and efficiently using interior-point
methods, and most of the theories related to convex opti-
mization have been already developed. Therefore, recog-
nizing or formulating a problem as a convex optimization
problem gives us a great advantage. In the texts [5, 6], authors have provided the readers with a very good background to develop a working knowledge on convex optimization to recognize, formulate, and solve convex optimization problems. For example, if we consider a nonconvex constrained optimization problem, the associated negative dual problem is always convex. Hence, in some situations, the original problem can be solved by using the dual problem which provides an easy environment to work with because of the convexity.

We have observed that currently available state-of-the-art distributed methods of solving optimization problems are gradient-based algorithms, subgradient-based algorithms, and their variants, such as ADMM [30, 32–38]. The gradient method is generally applied on unconstrained optimization problems. In 1970, Ramsay had studied gradient methods for optimizing nonlinear functions of several variables that cause difficulties when second derivative approaches are used [39]. In the recent study [40], Necif et al. have focused on solving a distributed convex optimization problem using “push-pull gradient methods.” They have given this interesting name as the agents in the problem network push the gradient information to the neighbors and the decision variable information is pulled by neighbors throughout the method. In [41], Calamai and Moré have studied the convergence properties of the projected gradient method for linearly constrained problems which are useful in large-scale problems. The projected gradient method is a variant of the gradient method which is used in constrained optimization.

The subgradient method can be considered as a generalization of the gradient method and is useful in optimizing nondifferentiable functions. In [9–12, 22, 42], subgradient methods are used to solve large-scale distributed problems that deals with the sum of a large number of convex local objective functions. References [24, 43–45] are some studies that have been focused on effects of constraints, and they have presented project subgradient algorithms to solve constrained optimization problems. In [44], Amini and Yousefian have studied a very important class of bilevel convex optimization problems that are often used for large-scale data processing in machine learning and neural networks. The authors in [45] have studied the binary iterative hard thresholding algorithm, a state-of-the-art recovery algorithm in one-bit compressive sensing which makes use of the projected subgradient method.

ADMM is also a well-suited method used in distributed convex optimization over large-scale networked systems arising in statistics and machine learning. The ADMM was first proposed by Gabay, Mercier, Glowinski, and Marrocco [46] in the mid-1970s. In the recent study [47], Xiao et al. have presented a distributed and scalable algorithm for managing the residential demand response programs using ADMM. They have shown through their simulation studies that the proposed method can reduce customers’ electricity bills and peak load. Authors in [48] have presented a distributed ADMM for solving the direct current dynamic optimal power flow with carbon emission trading problem. In [49], Hajinezhad and Shi proposed an algorithm related to ADMM to study a class of nonconvex nonsmooth optimization problems with bilinear constraints which are widely used in machine learning and signal processing application domains. The study [50] has presented a modified distributed ADMM to handle nonconvex optimization problems with discrete control variables.

4. The Gradient Method

Let us consider an unconstrained minimization problem as follows:

$$\text{minimize } f(x),$$

where $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is differentiable and $x \in \mathbb{R}^n$. Then, the gradient method to solve optimization problems of form (13) can be expressed by following iterative process, which starts from some initial point $x_0$:

$$x^{k+1} = x^k - \alpha_k \nabla f(x^k),$$

(14)

where $\alpha_k \geq 0$ is known to be the step size. The convergence of method (14) can be discussed under various considerations, using the theorems presented in [51].

**Theorem 1** (see [51]). Suppose that $\alpha_k = \alpha$ (a constant step size) in (14). Let $f(x)$ be differentiable on $\mathbb{R}^n$, $\nabla f$ is Lipschitz continuous with constant $L$, and let $f(x)$ be a strongly convex with constant $l$. Then, method (14) converges to a unique global minimum point $x^*$ with the rate of geometric progression when $0 < \alpha < 2/L$:

$$\|x^k - x^*\| \leq cq^k; \quad 0 \leq q < 1.$$  

(15)

Next, the following theorem shows the convergence of (14) for an even smaller class of functions.

**Theorem 2** (see [51]). Let $f(x)$ be strongly convex and twice differentiable. Suppose that

$$lI \leq \nabla^2 f(x) \leq lI; \quad l > 0, \forall x.$$  

(16)

Then, for $0 < \alpha < 2/l$,

$$\|x^k - x^*\| \leq \|x^0 - x^*\|q^k,$$

(17)

$$q = \max\{1 - \alpha l, 1 - \alpha l\} < 1.$$

Moreover, when $\alpha = 2/(L + l)$, $q$ is minimal and equal to $q^* = (L - l)/(L + l)$. The proofs of Theorem 1 and 2 are given in [51], and the convergence to a local minimum point of $f(x)$ is also discussed in the same text under Theorem 4 of Section 1.4. We discuss the convergence of the gradient method using a numerical example in the numerical results section (Section 8). In Section 8.1, our focus of discussion is the convergence results with the use of primal decomposition.

There are many early studies done on gradient methods [39, 41, 52, 53]. Authors in [53] had combined gradient methods with back propagation methods for neural networks to discuss the optimization of weights of multilayer
neural networks. In the study [52], authors have proposed two new step sizes for the classical-steepest descent method, where \(\alpha_k\) in method (14) is used as \(\alpha_k = \arg\min_x f(x^k - \alpha \nabla f(x^k))\). The most interesting fact regarding these new step sizes is that they require less computational effort than the classical-steepest descent method. However, these studies have not given enough attention and emphasis on distributed optimization techniques, which have become crucial to be analyzed in many application domains.

Some recent work that relies on gradient methods can be found in [8, 40, 54, 55]. In these studies, the gradient method has been applied with the use of distributed techniques. In [8], the authors have investigated fundamental properties of distributed optimization based on gradient methods, where gradient information is communicated using a limited number of bits. It is a well-known fact that message exchange between subsystems is a common phenomenon in distributed optimization settings. However, perfect message exchange between subsystems is not possible due to limited communication bandwidths between subsystems. Therefore, quantized information tends to be exchanged between users in networked systems, which led to the exploration of new findings on quantized distributed techniques. The study [8] is a very good initiative in this regard. This piece of work has studied a general class of quantized gradient methods where the gradient direction is approximated by a finite quantization set, to optimize a constrained convex optimization problem. Here, they have considered optimization problems of the form as follows:

\[
\min f(x) \\
\text{subject to } x \in \mathcal{X},
\]

where \(f\) is convex and differentiable with L-Lipschitz continuous gradient, \(x \in \mathbb{R}^n, \mathcal{X}\) is closed and convex set, and the optimal solution set \(\mathcal{X}^*\) is nonempty and bounded.

To solve problem (18), they have used the projected gradient method as follows:

\[
x^{k+1} = \left[x^k - \alpha_k d^k\right]_{\mathcal{X}},
\]

where \(d^k\) is quantized gradient information coded using limited number of bits. In this paper, authors have proposed two types of quantization schemes, namely, binary quantization and proper quantization.

(a) **Binary Quantization.** In this quantization scheme, the quantization set is taken as \(\mathcal{D} = \{1/\sqrt{n} (e_1, e_2, \ldots, e_n) | e_i \in [-1, 1]\}\), where \(d^k = \text{sign}(\nabla f(x^k))\). A convergence proof of method (19) was given under this binary quantization when \(\mathcal{X} = \mathbb{R}^n\) and \(\mathcal{X} = \mathbb{R}^n_+\). These convergence results are very important as they can deal with a dual problem of form (18) associated with equality and inequality constrained primal problems.

(b) **Proper Quantization.** When the above discussed binary quantization is used to solve TCP problems, the related quantized gradients are transmitted using \(n\) bits. There are many applications, where the dual problem is maintained by an individual coordinator [18, 19]. Therefore, it is worth seeking to analyze whether it is possible to use less number of bits than \(n\) when an individual coordinator exerts the problem. This fact motivates authors in [8] to discuss about the proper quantization. Here, we like to highlight the following two definitions they have used to establish their results.

**Definition 1** (see [8]). A finite set \(\mathcal{D}\) is a proper quantization for problem (18); if for every initialization \(x^0 \in \mathcal{X}\) in iterates (19), we can choose \(d^k \in \mathcal{D}\) and \(\alpha_k > 0, \forall k \in \mathbb{N}\), s.t. \(\lim_{k \to \infty} (\min_x \{x^* \in \mathcal{X} | \|x^* - x^k\|\}) = 0\).

**Definition 2** (see [8]). The finite set \(\mathcal{D}\) is a \(\theta\) - cover if \(\theta \in [0, \pi/2]\) and \(\forall g \in \mathcal{S}^n, \exists d \in \mathcal{D}\) s.t. \(\text{ang}(g, d) = \cos^{-1}(\langle g, d \rangle) < \theta\), where \(\mathcal{S}^n\) represents the unit sphere in \(\mathbb{R}^n\). It has been proved that \(\theta\) - cover \(\subseteq \mathcal{S}^n\) is a proper quantization for the problem class (18), and the minimal proper quantization is \(n + 1\) [8].

Authors in [54] have introduced two measures of communication complexity of dual decomposition, which help to identify the communication overhead required by limited communication networks. The first measure determines the smallest number of bits needed to find a solution within a given accuracy, whereas the second measure quantifies the best possible solution accuracy when a fixed amount of bits were communicated. Furthermore, in this same work, the authors have studied a quantization scheme (introduced as Primal-Feasible quantization scheme) which guaranteed primal feasibility at each iteration in their method.

5. The Subgradient Method

Subgradient method is basically used to minimize non-differentiable convex problems. Nondifferentiable or non-smooth functions are one important class of problems that arise in many applications of mathematical programming, such as game theory, multicriteria models, nonlinear programming problems, optimal control problems with continuous or discrete time, and integer and mixed integer programming problems [56]. Subgradient methods are first-order methods. Their performance highly depends on problem scaling and conditioning, whereas Newton’s method and interior-point methods are not dependent on problem scaling [57].

Before entering into the topic of subgradient methods, we would like to discuss about subgradients, which can be introduced as a generalized concept of gradients. When a function is nondifferentiable, the gradient of the function at nondifferentiable points cannot be found uniquely. Therefore, a well-defined way to express the slope of the function at those nondifferentiable points is required, mainly in optimization theory. Thus, getting a better understanding of subgradients is essential in the field of optimization theory. Reference [56] gives a very good exposition of the concept of subgradients, and it provides many important theoretical
aspects related to subgradients. Polyak’s text [51] and the text [6] of Bertsekas are two other good references that discuss subgradients and subgradient methods. Next, we will define a subgradient of a convex function.

**Definition 3.** A vector \( g \in \mathbb{R}^n \) is a subgradient of \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) at \( x \in \text{dom} \ f \) if for all \( y \in \text{dom} \ f, \)

\[
 f(y) \geq f(x) + g^T(y - x).
\]

The set of all subgradients of \( f \) at \( x \) is called the subdifferential of \( f \) at \( x \) and denoted by \( \partial f \). If \( f \) is differentiable, then its subgradient at \( x \) is unique and it is the gradient of \( f \) at \( x \).

### 5.1. The Basic Subgradient Method

We consider the same form of the unconstrained optimization problem (13) considered in Section 4. The objective function \( f(x) \) is still convex but not necessarily differentiable. Then, the subgradient method used to solve this problem can be given by the following iterative sequence starting at some initial point \( x^0 \):

\[
x^{k+1} = x^k - \alpha_k g^k,
\]

where \( x^k \) is the \( k \)th iterate, \( g^k \) is any subgradient of \( f \) at \( x^k \), and \( \alpha_k > 0 \) is the step size related to \( k \)th iteration. The subgradient method (21) can be considered as an extension of the gradient method (14). The difference is that, in each iteration, we use a subgradient \( g^k \) of the function \( f(x) \) at \( x^k \) instead of \( \nabla f(x^k) \) in (14). Moreover, the step size selection in the subgradient method is much different to the gradient method. In [57], Boyd has given five basic step size rules, namely, constant step size, constant step length, square summable but not summable, nonsummable diminishing, and nonsummable diminishing step length. From these five step size rules, we present three common ones as follows:

1. A constant step size, \( \alpha_k = \alpha \) is a positive constant and independent of \( k \).
2. Square summable but not summable: the step sizes satisfy \( \alpha_k \geq 0, \sum_{k=1}^{\infty} \alpha_k^2 < \infty, \sum_{k=1}^{\infty} \alpha_k = \infty \).
3. Nonsummable diminishing: the step sizes satisfy \( \alpha_k \geq 0, \lim_{k \to \infty} \alpha_k = 0, \sum_{k=1}^{\infty} \alpha_k = \infty \).

For example, \( \alpha_k = 1/k \).

Above choices for the step size \( \alpha_k \) do not depend on details computed during the subgradient algorithm. This fact differs from the step size rules found in standard descent methods, which uses current point and search direction. Good discussions on descent methods can be found in chapter 9 of [5] and chapter 8 of [58]. We can find many other choices for step size \( \alpha_k \) in addition to the choices mentioned above. In [51], Polyak has shown that the subgradient method (21) cannot converge rapidly under diminishing nonsummable step size rule. Therefore, the author has described another variant of the subgradient method, by introducing a different step size rule that depends on \( f^* \), the optimal value of \( f(x) \). We introduce this step size in Theorem 4.

Next, we discuss the convergence of the subgradient method (21) that relies on Boyd’s step size rules mentioned above. We use the following assumptions to discuss the convergence:

**Assumption 1.** Optimal set \( X^* \), the set of minimizers of problem (13) is nonempty

**Assumption 2.** \( \|g^k\| \) is bounded

**Assumption 3.** The number \( K \) s.t. \( \|x^0 - x^*\| \leq K \) is known, where \( x^* \in X^* \) and \( x^0 \) is the initial point of the algorithm

**Theorem 3** (see [57]). Let Assumptions 1, 2, and 3 hold and let \( f_{\text{best}} = \min_{i=1,...,k} f(x^i) \). Then, in method (21), the following inequality holds:

\[
f_{\text{best}} - f^* \leq \frac{R^2 + G^2 \sum_{i=1}^{K} \alpha_i^2}{2 \sum_{i=1}^{K} \alpha_i},
\]

where \( R \) s.t \( \|x^0 - x^*\| \leq R \) and \( G \) is s.t \( \|g^k\| \leq G \) for all \( k \).

The proof of Theorem 3 can be found in Section 3.2 of [57]. Using this theorem, one can show that the subgradient method converges within some range of the optimal value \( f^* \), for constant step size and constant step length. For other variants of the step size, square summable but not summable, nonsummable diminishing, and nonsummable diminishing step lengths, the subgradient method converges exactly to the optimal value without incurring any error. We discuss the convergence of the basic subgradient method empirically, in the numerical results section with the above presented three step size rules. In Section 8.2, we use a constrained optimization problem, and we dedicate our attention to discussing the convergence using dual decomposition. Next, we state the following theorem which gives the convergence of the subgradient method using Polyak’s step length.

**Theorem 4** (see [51]). Let the set of minimizers \( X^* \) of problem (13) (with nondifferentiable \( f \)) is nonempty and \( \alpha_k = (f(x^k) - f^*)/\|\partial f(x^k)\|^2 \). Then, in method (21), \( x^k \rightarrow x^* \in X^* \).

The proof of above theorem is given by Polyak in his book [51]. Now, we discuss and analyze some studies done on subgradient methods. In [22], authors have considered
a subgradient method to optimize a sum of convex objective functions corresponding to multiple agents. This work analyzes large-scale networked systems, where it is essential to design decentralized resource allocation methods, since the centralized solution methods are not suitable. This paper has considered a scenario where agents cooperatively minimize a common additive cost. The corresponding optimization problem can be posed as follows:

$$\min \sum_{i=1}^{m} f_i(x)$$
subject to $x \in \mathbb{R}^n$,  

where the function $f_i: \mathbb{R}^n \rightarrow \mathbb{R}$ represents the cost function of agent $i$, which is convex and not necessarily to be differentiable, and $x \in \mathbb{R}^n$ is the decision vector. To analyze this problem, authors have proposed the following subgradient method:

$$x^{(k+1)} = \frac{1}{m} \sum_{j=1}^{m} w^j(k) x^j(k) - \alpha^j(k) d^j(k), \quad i = 1, \ldots, m, \quad (26)$$

where $w^j_i$ represents the weight that agent $i$ assigns to the information $x^j$ received from a neighboring agent $j$ and the scalar $\alpha^j(k) > 0$ represents the step size used by agent $i$. The vector $d^j_i(k)$ is a subgradient of $i$’s objective function $f_i(x)$ at $x = x^{(k)}$. Next, to analyze the convergence of method (26), they have used a different representation of that method in a way that each iteration $x^{(k+1)}$ can be estimated using the information $w^j_i(s)$ and estimates $x^j(s)$, where $i, j = 1, \ldots, m$ and $s \leq k$. In this study, the authors have considered an unconstrained optimization problem, but in general, this problem can be viewed in a more advanced setting, in the presence of constraints. This fact motivates readers to extend this seminal work done by Nedić and Özdaglar to a different path of research, which will lead to a different line of convergence analysis. Furthermore, their model assumes that agents can exchange exact information, which is not possible in practice due to limited communication bandwidths. Therefore, the information is usually quantized before being sent, and it is considered that the quantization reduces the communication cost in networked control systems [59–61].

In [11], authors have considered the distributed subgradient method discussed in [22] and they have presented improved convergence results. Furthermore, they have shown that upper bounds for the difference between the estimated objective function value and the exact optimal value of the problem have a polynomial dependence on the number of agents $m$, by using results of their prior work [62]. We can view these bounds as improved versions of error bounds obtained in studies [22, 42], which involve exponential dependence on $m$. Moreover, the authors have studied the subgradient method when the communicated information is quantized to address the issue that perfect message exchange between agents cannot be performed. Some other works related to the same line of research are [9, 10, 12].

5.2. Projected Subgradient Method. Projected subgradient method is an extension of the basic subgradient method used in constrained optimization problems. Consider the optimization problem of the form

$$\min f(x)$$
subject to $x \in \mathcal{X}$,  

where $f$ and $\mathcal{X}$ are convex. Then, the projected subgradient method can be given by

$$x^{k+1} = \pi_{\mathcal{X}}[x^k - \alpha_k g^k], \quad (28)$$

where $g^k$ is any subgradient of $f$ at $x^k$. Convergence of method (28) can be attained under the same step size rules described under the basic subgradient method [57].

Authors in [43] have presented distributed algorithms to solve a constrained consensus problem and a constrained optimization problem. They have used a distributed projected subgradient method to solve the constrained optimization problem, which consist of minimizing a sum of convex local objective functions. They have shown that their method converges to the optimal solution with square summable but not summable step size rule. In [24], Madan and Lall have proposed two distributed projected subgradient methods to find an optimal routing flow to maximize the network lifetime in a partially and fully decentralized manner. In their solution, subgradient methods have been applied with their dual problem. We noticed that most of the studies performed on distributed optimization have used their original primal objective function in the optimization process. They have not shown much interest on duality theory, which provide many advantages in solving constrained optimization problems. Under these circumstances, Madan’s and Lall’s work [24] provides immense value addition to the study of distributed optimization.

6. Alternating Direction Method of Multipliers

ADMM is a simple but strong method that is used in distributed optimization [32]. ADMM is a variant of augmented Lagrangian and method of multipliers that uses the decomposability of dual ascent. In [32], augmented Lagrangian and method of multipliers are discussed under the following equality constrained optimization problem:

$$\min f(x)$$
subject to $Ax = b$, \quad (29)

where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, and $f$ is convex. Then, the augmented Lagrangian for problem (29) is given by

$$L_p(x, \lambda) = f(x) + \lambda^T (Ax - b) + \left(\frac{p}{2}\right) \|Ax - b\|^2, \quad (30)$$

where $p > 0$ is known as the penalty parameter. Then, the corresponding dual function is given by $g_p(\lambda) = \inf_x L_p(x, \lambda)$. The authors have used the gradient method to minimize negative $g_p(\lambda)$ with penalty parameter $p$ as the step size. The method of multipliers can be viewed as more
robust version of the dual ascent method, and it yields convergence under more general conditions than the dual ascent. However, “when f is separable, $L_p$ is not separable” is the fact that the authors in [32] have concerns with. When f is not separable, the minimization process cannot be continued in parallel, and hence, the method of multipliers cannot be used in dual decomposition. Therefore, an alternative way of observing problem (29) is needed, and consequently the ADMM has been introduced to address this issue. ADMM is a method well suited for distributed optimization settings that consist of large-scaled problems. In [32], authors have considered another variation of problem (29) as follows, to view it in separable form which has then led to the introduction of ADMM:

$$\begin{align*}
\text{minimize } & f(x) + g(y) \\
\text{subject to } & Ax + By = c,
\end{align*}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{q \times m}$, and $c \in \mathbb{R}^q$. Moreover, $f$ and $g$ are convex functions. Then, the distributed algorithm for ADMM can be given using Algorithm 1.

There are many early studies done on the method of multipliers and ADMM [63, 64]. Some recent studies done on ADMM can be found in [65–69]. In [65], Erseghe has proposed a fully distributed algorithm for optimal power flow using ADMM. In this paper, the author has introduced another variation on ADMM Algorithm 1 with assumptions such as $g(y) = 0$, where $y$ is contained in a linear space with associated orthogonal projector and also with certain assumptions on initial choices. In the study [66], authors have presented a decomposed solution approach with ADMM to solve a cost minimization problem, where the objective consists of energy and battery degradation cost. This work has used a modified version of ADMM, which helps to reduce the computations cost and ensures the stability of the solution. Most of the researchers including the ones mentioned above who have worked on ADMM have no concerns on noises that can be embedded in their models due to different types of errors occurring in practice, for an example, due to limited communication bandwidths. This fact motivates readers to work on this path with ADMM.

### 7. Distributed Optimization with Noise

The distributed methods for solving optimization problems can be applied in pure form only if errors and inaccuracies are fully avoided, which is hardly possible in the real world. As an example, errors or noises can occur due to inexact computation or measurement of subgradients and function values, sparsification [70], and quantization [8, 71]. The noise can be deterministic or random according to the behaviour of the application domain. Most of the real world problems consist of large-scale networked systems and mostly solve a common objective function interactively. In such situations, subsystems have to exchange their private information with neighboring subsystems during the optimization process. However, the subsystems may not be able to communicate exact information due to several reasons such as security measures and communication overheads. Therefore, it is very important to analyze distributed methods with noise imposed on the system.

#### 7.1. Distributed Methods with Noise for Optimizing Smooth Functions

In distributed methods for optimizing differentiable (smooth) functions, we always deal with a computation of the gradient, and instead of the exact value of the gradient $\nabla f(x^k)$, we may have it computed with error $\hat{\nabla f}(x^k) + r_k$:

$$\hat{\nabla f}(x^k) + r_k,$$

where $r_k$ is introduced to be the noise. In chapter 4 of [51], Polyak has discussed four types of most important classes of noise:

1. **Absolute deterministic noise**: $r_k$ is deterministic and satisfies the boundedness condition $\|r_k\| \leq \epsilon$.
2. **Relative deterministic noise**: $r_k$ is deterministic and satisfies the condition $\|r_k\| \leq c \|\nabla f(x^k)\|$. (31)
3. **Absolute random noise**: $r_k$ is random, independent, centered, and has bounded variance, $E[r_k] = 0$ and $E[\|r_k\|^2] \leq \sigma^2$.
4. **Relative random noise**: $r_k$ satisfies the condition $E[r_k] = 0$, $E[\|r_k\|^2] \leq \tau \|\nabla f(x^k)\|^2$. (32)

In the above classes of noise, $\epsilon, \sigma$, and $\tau$ represent positive constants. In the same text [51], the convergence of the gradient method (14) was discussed, where the gradient is computed with error as given in (32). Here, the convergence properties of the gradient method were analyzed under all four types of errors mentioned above, under the assumption that the objective function is strongly convex and with a gradient satisfying a Lipschitz condition.

Most of the related literatures available to solve optimization problems with the use of gradient like methods under the presence of noise were analyzed under boundedness assumptions on the objective function and the decision variable or show only $\lim_{k \to \infty} \inf \|\nabla f(x^k)\| = 0$ [72–75]. Authors in [55] discussed convergence results for the following method, by removing various boundedness conditions such as boundedness from below of $f$, boundedness of $\nabla f(x^k)$, or boundedness of $x^k$:

$$x^{k+1} = x^k + \alpha_k (s^k + w^k),$$

where $s^k$ represents a descent direction of a function $f: \mathbb{R}^n \to \mathbb{R}$ and $w^k$ is a deterministic or stochastic error. They first focus on the above method with deterministic error, with $w^k$ satisfying following conditions:

$$\|w^k\| \leq \alpha_k \left( q + p \|\nabla f(x^k)\| \right).$$

(34)
where $p$ and $q$ are some positive scalars. Then, the convergence of method (33) was obtained using following theorem.

**Theorem 5** (see [55]). Suppose that $s^k$ in method (33) is a descent direction satisfying for some positive scalars $c_1$ and $c_2$, and for all $k$,

$$
\begin{align*}
&c_1 \| \nabla f(x^k) \|^2 \leq -\langle \nabla f(x^k)^T s^k, s^k \rangle, \\
&\| s^k \| \leq c_2 (1 + \| \nabla f(x^k) \|).
\end{align*}
$$

(35)

Then, for $\alpha_k > 0$ with square summable but not summable step size rule, method (33) guaranteed to converge to the optimal solution.

Next, the authors have obtained convergence results for minimizing a sum of large number of functions using incremental gradient methods. Moreover, they have focused on stochastic gradient methods. In the recent study [68], authors have analyzed the convergence of distributed ADMM for consensus optimization in the presence of random error. They have presented lower and upper bounds on the mean squared steady state error of the algorithm when individual objective functions are strongly convex and when the gradients are Lipschitz continuous. Furthermore, authors have presented that steady state error of their noisy ADMM algorithm is bounded when they have a bounded random error and when individual objectives are proper, closed, and convex.

### 7.2. Distributed Methods with Noise for Optimizing Nonsmooth Functions

In chapter 5 of [51], Polyak has introduced the well-known subgradient method of optimizing nondifferentiable (nonsmooth) problems with noise,

$$
\begin{align*}
x^{k+1} &= x^k - \alpha_k s^k, \\
s^k &= \partial f(x^k) + r^k,
\end{align*}
$$

(36)

where $r^k$ is the noise imposed on the subgradient. The convergence results of the noisy subgradient method (36) have been discussed by the same author under the same classes of noises discussed in the previous subsection. In the early study [76], Polyak has studied minimization methods of a nonlinear function with nonlinear constraints when the values of the objective function, constraints, and gradients are computed with errors. In [77], authors have studied the effect of noise on subgradient methods for convex constrained optimization problems of form (27). They have discussed the convergence properties of the following projected subgradient method when the noise is deterministic and bounded:

$$
\begin{align*}
x^{k+1} &= \left[ x^k - \alpha_k \tilde{g}^k \right]_J,
\end{align*}
$$

(37)

where $\tilde{g}^k$ is an approximate subgradient of the form $\tilde{g}^k = g^k + r^k$, where $r^k$ is the noise and $g^k$ is an $\alpha_k$ subgradient of $f$ at $x^k$ for some $\alpha_k \geq 0$. Convergence properties of method (37) have been analyzed under three step size rules, namely, constant step size rule, diminishing step size rule, and dynamic step size rule which is given by

$$
\alpha_k = \gamma_k \frac{\bar{f}(x_k) - \bar{f}^*}{\| \tilde{g}_k \|_2}, \quad 0 < \gamma_k \leq 2, \forall k \geq 0,
$$

(38)

where $\bar{f}(x_k)$ is an error involved function value and $\bar{f}^*$ is a target value approximating the optimal value $f^*$. First, the convergence of method (37) has been obtained when the constrained set is compact. Secondly, the authors have analyzed their method using a convex objective function which has a sharp set of minima. The important results observed by authors were as follows: (a) in the first scenario, the method converges to the optimal value with some tolerance and (b) in the second scenario, the method converges exactly to the optimal value without any error.

It is very important to pay attention to the stochastic optimization processes since many practical problems cannot be viewed as deterministic structures. Some studies that paid attention to this particular area can be found in [76, 78]. Authors in [78] have studied stochastic quasigradient methods which allow solving optimization problems without calculating exact values of objectives and constraints. In [76], a general convex problem with noise was solved with assumptions as follows:

---

**Algorithm 1:** Alternating direction method of multipliers (ADMM).

Given initial $\lambda$.

Set $k = 1$.

**while** (stopping criterion)

1. $x$ minimization step:
   minimize $L_P(x, y^k, \lambda^k)$
   Let $x^* = \arg\min L_P(x, y^k, \lambda^k)$.

2. $y$ minimization step:
   minimize $L_P(x^*, y, \lambda^k)$
   Let $y^* = \arg\min L_P(x^*, y, \lambda^k)$.

3. Dual variable update:
   $\lambda^{k+1} = \lambda^k + p(Ax^* + By^* - c)$.
(i) The objective function and inequality constraint functions are convex continuous
(ii) Feasible set is a convex closed bounded set
(iii) Slater condition holds
(iv) All noises are with mean zero with bounded variance and are independent at different points

8. Numerical Results

In this section, we discuss the convergence of the gradient method, subgradient method, and ADMM empirically by using some numerical examples.

8.1. Example 1 (Gradient Method: Primal Decomposition).

Here, we consider an unconstrained minimization problem with two users as follows:

$$\text{minimize } f(x_1, x_2, y) = f_1(x_1, y) + f_2(x_2, y),$$

where

$$f_1(x_1, y) = \left[ x_1^T \ y^T \right] A_1 \left[ x_1^T \ y^T \right]$$

and

$$f_2(x_2, y) = \left[ x_2^T \ y^T \right] A_2 \left[ x_2^T \ y^T \right]$$

with $x_1, x_2 \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$. Here, $A_1 \in \mathbb{R}^{(n_1 + n_2) \times (n_1 + n_2)}$ and $A_2 \in \mathbb{R}^{(n_1 + n_2) \times (n_1 + n_2)}$ are positive definite matrices. We use primal decomposition and analyze the convergence of the gradient method (14) for this problem with the use of Theorem 1. The subproblems related to (39) can be given as follows:

**Subproblem 1:** $S_1(y) = \min_{x_1} \left[ x_1^T \ y^T \right] A_1 \left[ x_1^T \ y^T \right]$

**Subproblem 2:** $S_2(y) = \min_{x_2} \left[ x_2^T \ y^T \right] A_2 \left[ x_2^T \ y^T \right]$

Then, the master problem corresponding to (39) is given by

$$\text{minimize } S(y) = S_1(y) + S_2(y).$$

Analytically, by solving the subproblems, we can show that $S_1(y)$ is quadratic as $S_1(y)$ and $S_2(y)$ are quadratic. Moreover, $S_1(y)$ and $S_2(y)$ are strongly convex since $A_1$ and $A_2$ are positive definite. Hence, $S(y)$ is also strongly convex and $\nabla S(y)$ is Lipschitz continuous. Therefore, we can apply Theorem 1 to solve problem (40) using the gradient method (14). We use Algorithm 2 to solve (40). In this algorithm, at each iteration, the gradient update is given by $\nabla S_1(y^{k+1}) = \nabla S_1(y^{k}) + \nabla S_2(y^{k})$, where $\nabla S_1(y^{k}) = A_1^T x_1^k + A_2^T y^{k} + 2A_1^T x_1^k + 2A_2^T y^{k}$ and $\nabla S_2(y^{k}) = A_2^T x_2^k + A_3^T y^{k} + 2A_2^T x_2^k + 2A_3^T y^{k}$.

First, we illustrate our results with scalar valued primal variables $x_1, x_2,$ and $y$ ($n_1 = n_2 = 1$ case) for different values of constant step sizes $\alpha$. Figure 2 shows the convergence of $y^k$ with $\alpha = 0.001, \alpha = 0.01, \alpha = 0.1,$ and $\alpha = 0.5$. Next, we show the convergence results for different dimensions of the complicating variable $y$ with $x_1 \in \mathbb{R}^{10}$ and $x_2 \in \mathbb{R}^{10}$. Figure 3 shows the convergence of the residuals $\|y^k - y^{k-1}\|$ with step size $\alpha = 0.001$, for $y \in \mathbb{R}, y \in \mathbb{R}^2, y \in \mathbb{R}^3$, and $y \in \mathbb{R}^{10}$, where $y^*$ represents the optimal value of $y$. We present Figure 4, which indicates log values of $\|y^k - y^{k-1}\|$, to analyze the convergence of residuals when they approach to zero. For this same set of dimensions of $y$ with same step size, the convergence of $y^k$ is shown under Figure 5. Moreover, Figure 6 shows that the primal variable iterates $x_1^k$ and $x_2^k$ converge exactly to their optimal solutions using $\alpha_k = 0.001$ and $y \in \mathbb{R}$.

8.2. Example 2 (Subgradient Method: Dual Decomposition).

Here, we focus on a problem which is not quadratic. We consider the problem in the following form with two users:

$$\text{minimize } f(x_1, y_1, x_2, y_2) = f_1(x_1, y_1) + f_2(x_2, y_2)$$

subject to $y_1, y_2 \in \mathbb{X}, x_2 \in \mathbb{X}, y_1, y_2 \in \mathbb{Y},$

where

$$f_1(x_1, y_1) = \cosh(a_1^T x_1) + a_2^T x_1$$

and

$$f_2(x_2, y_2) = \cosh(b_1^T y_2) + b_2^T y_2$$

with $x = (x_1, y_1), y = (x_2, y_2), \mathbb{X} \subseteq \mathbb{R}^r, \mathbb{Y} \subseteq \mathbb{R}^s,$ and $a_1, a_2, b_1, b_2 \in \mathbb{R}^{n_1 \times n_2}$. Here, we intend to solve this problem in a fully distributed manner using dual decomposition. We implement our results for $n_1 = n_2 = 1$ (scalar valued variables). We consider $\mathbb{X} = [-1, 0], \mathbb{X} = [1, 2], \mathbb{Y} = [-2, 1], a_1 = [1, 1]^T, a_2 = [3, -2]^T, b_1 = [1, 1]^T,$ and $b_2 = [-2, 5]^T$. The dual function corresponding to the primal problem (41) is given by

$$g(\lambda) = \inf_{x_1 \in \mathbb{X}, y_1 \in \mathbb{Y}} (f_1(x_1, y_1) + f_2(x_2, y_2) + \lambda^T (y_1 - y_2)),$$

and we use corresponding subproblems in dual decomposition as follows:

**Subproblem 1:** $g_1(\lambda) = \inf_{x_1 \in \mathbb{X}, y_1 \in \mathbb{Y}} (f_1(x_1, y_1) + \lambda^T y_1)$

**Subproblem 2:** $g_2(\lambda) = \inf_{x_2 \in \mathbb{X}, y_2 \in \mathbb{Y}} (f_2(x_2, y_2) - \lambda^T y_2).$

Then, the dual problem corresponding to the primal problem (41) is given by maximizing $g_1(\lambda) + g_2(\lambda)$. We know that $g(\lambda)$ is always concave (see chapter 05 of [5]). We have obtained the graph of $g(\lambda)$ as given in Figure 7. This figure also confirms the concavity of $g(\lambda)$. Moreover, this figure shows that $g(\lambda)$ is non-differentiable as it has a sharp point around $\lambda = 5$. Hence, $-g(\lambda)$ is convex and non-differentiable, and therefore we use subgradient method (21) to minimize $-g(\lambda)$ using Algorithm 3.

We analyze the convergence results of the subgradient method using Theorem 3 discussed under Section 5. Therefore, we have to check whether Assumptions 1–3 used in Theorem 3 hold for our particular problem considered here. Figure 7 shows that there exists an optimal solution $\lambda^*$.
Given initial $y, y^0$.
Set $k = 0$.

**while** (stopping criterion)

1. $x_1$ and $x_2$ minimization steps:
   - Step 1: $x_1^k = \arg\min_{x_1} A_1 x_1^T y_{kT} + A_1 x_1^T y^k$.
   - Step 2: $x_2^k = \arg\min_{x_2} A_2 x_2^T y_{kT} + A_2 x_2^T y^k$.

2. gradient information update:
   $$\nabla S(y^k) = A_{11} x_1^k + A_{13} x_2^k + 2 A_{14} y^k + A_{21} x_1^k + A_{23} x_2^k + 2 A_{24} y^k.$$

3. $y$ variable update:
   $$y^{k+1} = y^k + \alpha \nabla S(y^k)$$

**Algorithm 2: Gradient method: primal decomposition.**

---

**Figure 2:** Convergence of $\log_{10} ||y^k - y^*||$ using different constant step sizes in the gradient method with primal decomposition. The figure shows that slow convergence for relatively small step sizes.

---

**Figure 3:** Convergence of the residuals $||y^k - y^*||$ for different dimensions of $y$ when the primal problem (39) is solved using the gradient method followed by primal decomposition. The figure shows that $||y^k - y^*|| \to 0$ as the iteration number increases. This shows that the convergence of $y^k$ to $y^*$ is guaranteed even for high dimensional complicating variables.

---

In our Algorithm 3, both users solve their subproblems separately and find optimal primal variables locally at each iteration. Next, they exchange their information $y^k_1$ and $y^k_2$ with each other and update the dual variable individually. In general, their iterates $y^k_1$ and $y^k_2$ are not feasible. Therefore, at each iteration, they agreed to have a feasible solution as $y^k = (y^k_1 + y^k_2)/2$. Next, by using these primal variable iterates and updated dual variable $\lambda^k$, user 1 and user 2 can compute $g_1(\lambda^k)$ and $g_2(\lambda^k)$, respectively. Then, $g(\lambda^k) = g_1(\lambda^k) + g_2(\lambda^k)$ can be calculated. This is always a lower bound on $f^\star$, the optimal value of the primal problem [5]. Moreover, at each iteration, users can compute two upper bounds on $f^\star$ as follows [14]:

$$u(x_1^k, x_2^k, y^k) = f_1(x_1^k, y^k) + f_2(x_2^k, y^k),$$

$$b(y^k) = b_1(y^k) + b_2(y^k), \quad (44)$$
where $b_1(y^k) = \min_{x_1 \in X_1} f_1(x_1, y^k)$ and $b_2(y^k) = \min_{x_2 \in X_2} f_2(x_2, y^k)$. In [14], $w$ and $b$ are defined as the worst bound and the better bound. Worst bound represents the primal objective function values evaluated at each iteration using feasible points $(x_1^k, y^k)$ and $(x_2^k, y^k)$. Better bound can be obtained by replacing $y_1^k$ and $y_2^k$ with $y$ and then solving subproblems involved with related primal decomposition structure of (41). Figure 10 shows the convergence of $g(\lambda^k)$, better bound, and worst bound using constant step size rule $\alpha_k = 0.1$ and scalar valued primal
variables. Here, we can observe that for this particular problem, the lower bound $g(\lambda)$ and two upper bounds converge exactly to $f^*$. 

8.3. Example 3 (ADMM). Here, we first discuss the robustness of ADMM compared with the gradient method. Let us consider the following linear programme:

$$
\begin{align*}
\text{minimize} & \quad a^T x \\
\text{subject to} & \quad Ax = b \\
& \quad Bx + Cy = 0,
\end{align*}
$$

(45)

where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ are decision variables of the problem, $a \in \mathbb{R}^n, A \in \mathbb{R}^{m_1 \times n}, B \in \mathbb{R}^{m_2 \times n}, C \in \mathbb{R}^{m_3 \times n}$ with $m_1, m_2 < n$ and $b \in \mathbb{R}^{m_1}$ is a constant vector. Suppose that the set of solutions of (45) is nonempty.

The dual function $g(\lambda)$, where $\lambda = \left[ \lambda_1^T, \lambda_2^T \right]^T$ with $\lambda_1 \in \mathbb{R}^{m_1}$ and $\lambda_2 \in \mathbb{R}^{m_2}$, for problem (45) is given by

$$
g(\lambda) = \inf_{x \in \mathbb{R}^n, y \in \mathbb{R}^n} \left( a^T x + \lambda_1^T (Ax - b) + \lambda_2^T (Bx + Cy) \right)
$$

$$
= \inf_{x \in \mathbb{R}^n} \left( a + A^T \lambda_1 + B^T \lambda_2 \right)^T x + \inf_{y \in \mathbb{R}^n} \left( \lambda_2^T Cy - \lambda_1^T b \right).
$$

(46)

Then, analytically we can obtain

$$
g(\lambda) = \begin{cases} 
-\lambda_1^T b; & a + A^T \lambda_1 + B^T \lambda_2 = 0 \text{ and } C^T \lambda_2 = 0, \\
-\infty; & \text{otherwise}.
\end{cases}
$$

(47)

Next, the dual problem is given by

$$
\text{maximize} \quad g(\lambda).
$$

(48)

Here, we can easily observe that the optimal value of the dual problem (48) is $-\lambda_1^T b$, which is attained when $a + A^T \lambda_1 + B^T \lambda_2 = 0$ and $C^T \lambda_2 = 0$. Usually we use following subproblems when we use the gradient method to solve (48):

**Algorithm 3: Subgradient method: dual decomposition.**
Subproblem 1: \( g_1(\lambda_1, \lambda_2) = \inf_{x \in \mathbb{R}^n} \left( a + A^T \lambda_1 + B^T \lambda_2 \right)^T x \),

Subproblem 2: \( g_2(\lambda_1, \lambda_2) = \inf_{y \in \mathbb{R}^n} (\lambda_2^T Cy - \lambda_1^T b) \).

Algorithm 4 represents the corresponding gradient algorithm.

We can observe that \( x \) and \( y \) minimization steps (Algorithm 4) given in Algorithm 4 cannot proceed for any arbitrarily chosen \( \lambda \) as \((a + A^T \lambda_1^k + B^T \lambda_2^k)^T x \) and
Given initial \( \lambda^0, \lambda^0 \) = \((\lambda_1^0, \lambda_2^0)\).
Set \( k = 0 \).

while (stopping criterion)

(1) Primal variables minimization steps:
   Step 1: \( x^k = \arg\min_{x \in \mathbb{R}^n} (a^T x + \lambda_1^k (Ax - b) + \lambda_2^k (Bx + Cy) + (p/2)\|Ax - b\|^2 + (p/2)\|Bx + Cy\|^2) \)
   Step 2: \( y^k = \arg\min_{y \in \mathbb{R}^m} (\lambda_1^k (Ax^k - b) + \lambda_2^k (Bx^k + Cy) + (p/2)\|Ax^k - b\|^2 + (p/2)\|Bx^k + Cy\|^2) \)

(2) Dual variable update:
   \( \lambda_1^{k+1} = \lambda_1^k + \alpha (Ax^k - b) \)
   \( \lambda_2^{k+1} = \lambda_2^k + \alpha (Bx^k + Cy^k) \)

Algorithm 4: Gradient algorithm to solve (48).

\[
\begin{align*}
\text{Given initial } \lambda, \lambda^0 & = (\lambda_1^0, \lambda_2^0), \text{ and initial } x, y, \lambda^0. \\
\text{Set } k & = 0. \\
\text{while (stopping criterion)} \\
(1) & \text{ Primal variables minimization steps:} \\
\text{Step 1: } x^{k+1} & = \arg\min_{x \in \mathbb{R}^n} (a^T x + \lambda_1^k (Ax - b) + \lambda_2^k (Bx + Cy) + (p/2)\|Ax - b\|^2 + (p/2)\|Bx + Cy\|^2) \\
\text{Step 2: } y^{k+1} & = \arg\min_{y \in \mathbb{R}^m} (\lambda_1^k (Ax - b) + \lambda_2^k (Bx + Cy) + (p/2)\|Ax - b\|^2 + (p/2)\|Bx + Cy\|^2) \\
(2) & \text{ Dual variable update:} \\
\lambda_1^{k+1} & = \lambda_1^k + \alpha (Ax^k - b) \\
\lambda_2^{k+1} & = \lambda_2^k + \alpha (Bx^k + Cy^k) \\
\end{align*}
\]

Algorithm 5: Example 3: (ADMM).

\[
\begin{array}{l}
\lambda_2^k Cy - \lambda_1^k b \text{ are unbounded below. Hence, the gradient method fails to solve (48), and therefore the linear programme (45) also cannot be solved. However, the interesting fact is that ADMM solves this problem without any issue, showing its robustness compared with the gradient method.} \\
\text{To solve (48) using ADMM, we consider the augmented Lagrangian as follows:} \\
L(x, y, \lambda) = a^T x + \lambda_1^k (Ax - b) + \lambda_2^k (Bx + Cy) + \frac{p}{2}\|Ax - b\|^2 + \frac{p}{2}\|Bx + Cy\|^2, \\
\end{array}
\]

where \( p \) represents the penalty parameter. Then, the corresponding dual function is given by \( g(\lambda) = \inf_{x \in \mathbb{R}^n} L(x, y, \lambda) \). Next, we maximize \( g(\lambda) \) by using Algorithm 5. In this algorithm, \( \alpha \) represents a suitably chosen step size. Here, we discuss the convergence of iterates (Algorithm 5) with \( a = (-1, -3, -4)^T, \\
A = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 3 \end{bmatrix}, \\
B = \begin{bmatrix} -2 & 1 & 3 \\ 5 & 2 & -2 \end{bmatrix}, \\
C = \begin{bmatrix} -1 & 4 & -2 \\ 1 & 5 & -3 \end{bmatrix}, \\
b = [4, 5]^T. \\
\]

We choose \( p = \alpha = 0.1 \). Figure 11 shows that our method guarantees the convergence of the dual variable \( \lambda \) exactly to its optimal value \( \lambda^* \). Convergence of the dual function iterates \( g(\lambda^k) \) and the objective function iterates \( f(x^k) = a^T x^k \) is given in Figure 12, and it shows that the both dual function and objective function converge exactly to the optimal value \( f^* = -6 \) of our primal problem (45).
9. Conclusions

Centralized methods are hardly being used or applied as they are not suitable or they fail to deploy in many optimization settings due to the high demanding necessity of distributed techniques among large-scale networked systems. Therefore, an attempt has been made by this paper to discuss the most important methods that currently exist to solve distributed optimization problems. A detailed analysis on gradient like methods, subgradient methods, and ADMM has been presented with numerical results. Gaps in previous studies that need to be developed to enhance the process of distributed optimization over networked systems have been discussed under each section. Here, we summarize the areas

![Figure 11: Convergence of the dual variable iterates $\lambda^k$ in ADMM. The figure shows the log values of $||\lambda^k - \lambda^*||$ with iteration number. A better convergence up to $10^{-6}$ tolerance is achieved within 100 iterations.](image1)

![Figure 12: Convergence of the dual function and objective function iterates in ADMM. Horizontal line represents the optimal value $f^* = -6$ of the primal problem (45). The figure shows the iterates almost converge to $f^*$ within 100 iterations.](image2)
in which future research can be conducted in distributed optimization.

(i) Many studies have shown their interest to solve distributed problems using primal measures. Therefore, more theoretical studies related to duality need to be established to make use of the advantage of optimizing more general (nonconvex) distributed problems.

(ii) Methods of finding primal solutions from the dual under more relaxed assumptions are critical, as the dual measures do not converge to primal measures in general.

(iii) Distributed methods over limited communications between networked systems need to be analyzed in depth, and related proper quantization schemes that guarantee the convergence of corresponding distributed methods should be identified.

(iv) Inexact message exchange between subsystems due to limited communication bandwidths is common in distributed optimization. Consequently, analysis of error-based distributed methods is essential over many distributed application domains.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors thank Dr. Chathuranga Weeraddana for his continuous assistance and valuable comments given during this study. The authors would also like to show their gratitude to Mr. Susil Palihakkara for his feedback and comments provided to improve the presentation of this paper.

References

[23] Q. Zhang, K. Dehghanpour, Z. Wang, F. Qiu, and D. Zhao, "Multi-agent safe policy learning for power management of


